6C－1

ワークステーションにおける高速プロトコル処理 (0–copy architecture) の実装と性能評価 II

北村浩　西田竹志
日本電気株式会社 C&C 研究所
e-mail: kitamura@nwk.cl.nec.co.jp

1 はじめに

近年の100Mbps以上の高速LANの出現や、高速なCPU環境のもとで、従来のデータとは質量共に異なる音声や画像などの時間的制約の厳しいマルチメディア通信アプリケーションが増加しており、End-to-Endで実際のスループットとして高い通信性能を持つワークステーション(WS)の出現が急務となってきている。しかし、実際にはホストの実効スループットは伸びてきておらず、高速LANの通信速度の半分にも満たない通信性能しか出せない。通信性能が上がらない原因は、通信処理の中で主記憶装置上でのデータのコピー処理の部分にあることが性能評価より分かっている。我々はその解決方法として、WS内部で通信データのコピー処理を行なわない構造（**0-copy architecture**）を提案し、その実装についても既に報告した。

本稿では、改良を加えて安定度を増した実装で詳細な性能評価を行ない、**200Mbps**以上の通信性能を示した実装方式の改良点と性能評価について報告する。

2 今回の実装における改善点

2.1 SVR4.2 への移植 これまでSVR4ベースのOSの上に0–copy architecture の実装を行なってきた。今回はSVR4.2ベースOSの上で実装を行なうことにより、最新のアプリケーションなどに対応出来る機構にすると同時に、これらを通して0-copy architectureが、OS間での移植が非常に容易であることを証明した。

2.2 UDP への対応 これまでは TCP による接続だけを対象としていた。今回はマルチメディアや同報通信で利用の機会の多い UDP によるデータグラム型の通信に対しても 0-copy architecture の機能を利用するためのインタフェースを提供した。

2.3 UIO への機能の移動 これまでは 0-copy architecture を実現するために、入出力をポインタの受渡しに変換する仕組みをストリーム機構のモジュール (Exchange モジュール) として実現していた。しかし、この実装方法はいくつかの問題点があり、今回この変換の仕組みをストリームモジュールとしてではなく、ユーザ空間とカーネル空間の間でデータのコピーを行う部分 (UIO: User IO) に実装を行うことにより改善を行なった。以下がこの改善により生じる効果である。

- プロセスが実際にデータの読み込みを行うまで、入力データはカーネル内に特定のプロセスと対応付けられずに保持されるため、fork した子プロセスがそのデータを読み込むことが可能。
- プロセスの挙動とは非同期にアドレス空間に対するメモリのマッピングが行われていた。また、Exchangeモジュールがマッピングを変更するまでは、プロセスのアドレス空間にマッピングされたままであった。本改善により、データの読み込みを行うまでプロセスのアドレス空間のマッピングは変更されず、データ送信のシステムコールの実行と同時にプロセスのアドレス空間からデータが切り放されるため、プロセスがアドレス空間の割り当てを管理することが可能。
- 到着したパケットを1つずつExchangeモジュールの中で処理し、ユーザ空間へのマッピングを行っていたため、アドレス空間の検索が頻繁に起き、性能の低下を招いていた。本改善で、UIO レベルでマッピングを行うことにより、ストリームヘッドに溜った複数のパケットをまとめて処理することが可能になるため、検索のオーバーヘッドが軽減。

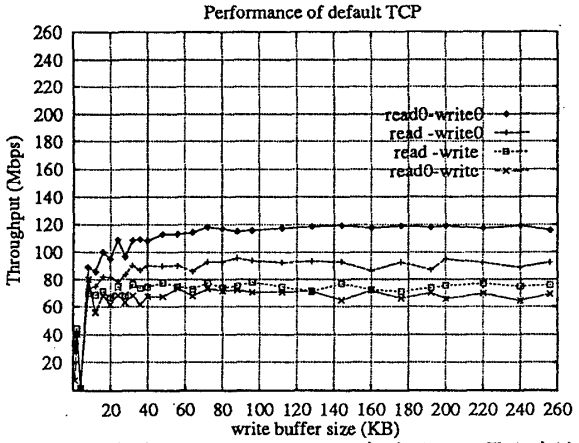
3 性能評価

一回の測定に付き128MBのデータを通信するソケット間通信プログラムを用いて時間を測定し性能評価を行なった。物理的制約が比較的少ないLLC層で折り返し同一の装置内で送受信両方を行なうループバック **I/F** を通信媒体として用いた。

Implementation and Performance Evaluation of High Speed Protocol Processing (0–copy architecture) in Workstaion II
Hiroshi KITAMURA　Takeshi NISHIDA
C&C Research Laboratories, NEC Corporation

測定結果と考察 送受信各々に、通常の方法と 0-copy arch. を用いた方法があるので、4 通りの組合せが可能であり、全ての組合せについて評価を行なった。*read0*や*write0*のように、関数名に 0 を含むものが 0-copy arch. を用いた通信である (各々受信、送信である)。

3.1 通常処理の TCP での通信

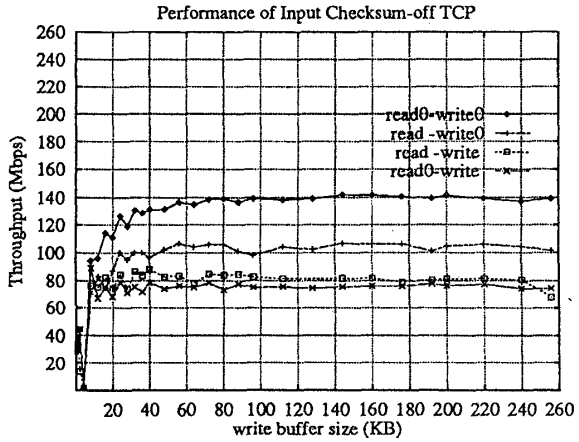


0-copy 通信の *read0–write0*の組合せが、最も高速であり、普通の *read–write*の組合せより、**54%** 向上し、**120Mbps** のスループットに達している。

受信が 0-copy である *read0–write*の組合せは、送信が普通であるために page 境界にならないため、*read0*で読み出す際 page 境界合せる copy 処理を行ないながら data を読んでいる。そのため、*read–write*の組合せより page mapping 処理の分だけ遅くなる。

3.2 入力 checksum 処理 off の TCP での通信

copy 処理に次いで重い処理は、TCP 層での checksum 処理であることが分かっている。そこで、kernel を改造し入力された data の checksum 値が正しいか調べる処理を止めた状態での通信結果である。

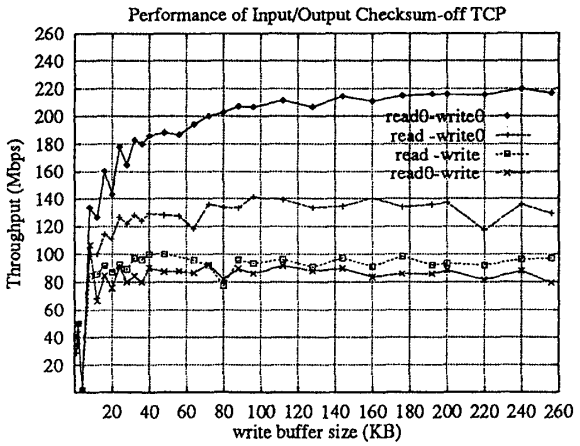


通常の TCP 通信と同じ傾向だが、*read0–write0*の組合せの性能が更に向上し、*read–write*の組合せより **75%** 向上し、**140Mbps** のスループットに達している。

入力 checksum 処理を止めることにより、*read–write*の組合せの性能向上が僅かであるのに対し、*read0–write0*の組合せでは、大きな性能向上が見られる。これは、メモリアクセスする処理である checksum 処理が 0-copy 通信で占める割合が一段と高いことを示している。

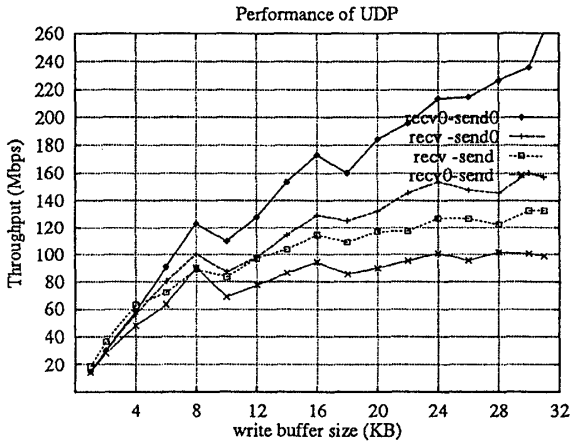
3.3 入出力 checksum 処理 off の TCP での通信

全ての checksum 処理を止めた状態での通信結果である。



更に顕著な性能向上が見られ、*read0–write0*の組合せで **218Mbps** のスループットを達成している。

3.4 通常処理の UDP での通信



TCP での通信より高速になっている。8KB の page size 毎に性能に差が現れるのが分かる。

4 まとめ

0-copy architecture phase II の実装方法の改善点の概略と、この構造を用いた通信での性能評価について報告した。実測値として **218Mbp** の高い性能を示しており、0-copy architecture の有用性が証明された。今後は ATM LAN などの実際の通信媒体を用いた通信で利用していく予定である。